Logistic Regression and Softmax Regression

Softmax regression is a generalization of logistic regression for multiclass

(Softmax regression is a 2-layer neural network)

logistic regression is softmax regression when $K=2$

Proof:

logistic regression states that:

$$P(Y=1|X,\theta_1) = \underbrace{\theta(X,\theta_1)}_{\uparrow \text{sigmoid}} = \frac{1}{1+e^{-(\theta_0+\theta_1^T x)}} = \frac{e^{\theta_0+\theta_1^T x}}{1+e^{\theta_0+\theta_1^T x}} = \frac{e^{\theta_{10}+\theta_1^T x}}{e^{\theta_{00}+\theta_0^T x}+e^{\theta_{10}+\theta_1^T x}}$$

$1 = e^{\theta_{00}+\theta_0^T x}$ if $Y=0$, $\uparrow$

(logistic regression only has 1 weight vector)

$$P(Y=0|X,\theta_0) = 1 - P(Y=0|X,\theta_1)$$

$$= 1 - \frac{e^{\theta_{10}+\theta_1^T x}}{e^{(\theta_{00}+\theta_0^T x)}+e^{(\theta_{10}+\theta_1^T x)}}$$

$$= \frac{e^{(\theta_{00}+\theta_0^T x)}+e^{(\theta_{10}+\theta_1^T x)}}{e^{(\theta_{00}+\theta_0^T x)}+e^{(\theta_{10}+\theta_1^T x)}} - \frac{e^{(\theta_{10}+\theta_1^T x)}}{e^{(\theta_{00}+\theta_0^T x)}+e^{(\theta_{10}+\theta_1^T x)}}$$

$$= \frac{e^{(\theta_{00}+\theta_0^T x)}}{e^{(\theta_{00}+\theta_0^T x)}+e^{(\theta_{10}+\theta_1^T x)}}$$

roughly:

$$P(Y=1|X,\theta) = \frac{e^{\theta_1^T x}}{e^{\theta_0^T x}+e^{\theta_1^T x}}$$

$$P(Y=0|X,\theta) = \frac{e^{\theta_0^T x}}{e^{\theta_0^T x}+e^{\theta_1^T x}}$$

---

same

Softmax regression states that:

for $K=2$:

$$z_1 = \theta_1^T x$$
$$z_2 = \theta_2^T x$$

Softmax is:

$$\frac{1}{\sum_{j=1}^K e^{z_j}} \begin{bmatrix} e^{z_1} \\ \vdots \\ e^{z_K} \end{bmatrix} = \frac{1}{\sum_{j=1}^K e^{w_j^T x}} \begin{bmatrix} e^{w_1^T x} \\ \vdots \\ e^{w_j^T x} \end{bmatrix}$$

(activation function)

$$W = \begin{bmatrix} - & w_1^T & - \\ - & w_2^T & - \\ & \vdots & \\ - & w_K^T & - \end{bmatrix}$$

$$P(Y=0|X,\theta) = \frac{e^{\theta_0^T x}}{e^{\theta_0^T x}+e^{\theta_1^T x}}$$

$$P(Y=1|X,\theta) = \frac{e^{\theta_1^T x}}{e^{\theta_0^T x}+e^{\theta_1^T x}}$$

How do we apply gradient descent / optimize the parameters? We have to use (backpropagation)

Cross Entropy Loss function (for K classes) is the softmax loss: (Loss function)

for 2 classes

$$L(W) = -\left[\sum_{i=1}^m \sum_{k=1}^K 1\{y_i=k\} \log\left(\frac{e^{w_k \cdot x_i}}{\sum_{j=1}^K e^{w_j \cdot x_i}}\right)\right]$$

m examples, K classes, $1\{y_i=k\}$ is an indicator variable when the expression inside is true. (can be implemented with one-hot encoding with indicator function.

cross-entropy and log likelihood loss are basically the same since cross-entropy explicitly computes a softmax calculation and does one-hot encoding, which can be done with an indicator function, and passes it into the log likelihood function.